Remarks

The present response is to the Office Action mailed in the above referenced case on October 04, 2007. Claims 1-28 are standing for examination.

Claim Objections

4. Claims 1-11 are objected to because of the following informalities: Claim 1 does not show the current status of the claim language. Specifically claim 1 is currently missing, "executing on a server", previously amended into the preamble of the claim. Therefore the Examiner is interpreting claim 1 as previously recited in the copy of the claims submitted 02/26/07. Appropriate correction is required.

Applicant's Response

Applicant herein corrects claim 1 to reflect the amendment made to the claim submitted on 2/26/07, as required by the Examiner.

Rejection 35 U.S.C. 101

6. Claims 1-11 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter.

Examiner's rejection

Claims 1-11 recite software program, which imparts functionality when employed as a computer component. Thus the software program of claims 1-11 is considered functional descriptive material. Functional descriptive material per se is not statutory. Functional descriptive material must be claimed in combination with an appropriate computer readable medium to enable the functionality to be realized with the computer. Thus claims 1-11 are rejected under 35 U.S.C. 101 for failing to fall within a statutory category and for failing to be structurally and functionally interconnected with the software in

such a manner to, in and of itself, enable any usefulness to be realized. Appropriate correction is required.

Applicant's Response

Applicant herein amends claim 1 to reflect software "executing on a server". Therefore, the rejection is overcome.

Merit Rejection 35 U.S.C. 103(a)

8. Claims 1-28 are rejected under 35 U.S.C. 103(a) as being unpatentable over DaCosta et al (US-6,826,553 11/30/04) in view of Weinberg et al (US-6,360,332 03/19/02).

Examiner's rejection

-In regard to substantially similar independent claims 1 and 12, DaCosta teaches an application for enabling automated notification of applied structural changes to electronic information pages on a network comprising:

an interface for enabling users to build and modify network navigation and interaction templates using functional logic blocks for automatically navigating to and interacting with interactive electronic information pages on the network (column 2, lines 11-30; column 5, lines 30-67)(Figs. 1 & 7);

a navigation interface for integrating the software application to a proxy navigation system for periodic execution of the templates (column 5, lines 19-20: "automatically repeat these steps in a scheduled manner or when requested");

a change notification module for indicating a navigation and interaction routine has failed and for creating a data file associated with the failed routine (column 18, lines 43-67: "it is known the script has failed. . . and proper notifications sent to individuals or entities responsible for the operation of the failing script by email. . . for example"; column 19, lines 1-15); and

sending proper notifications of the failed script to the developer upon failure of the script (column 6, lines 9-13 & 35-41; column 18, lines 543-67; column 19, lines 1-15). DaCosta does not specifically teach storing the data file in a data repository with a point-of-failure indication, parameters associated with the failed routine, and an identifier of the associated electronic information page subjected to the navigation. Weinberg teaches storing the data file (column 2, lines 3 9-40; column 6, lines 19-22), wherein the application periodically submits test navigation and interaction routines (column 6, lines 19-22), and upon failure of the routine, creates a data file (column 2, lines 39-40; column 3, lines 29-43; column 6, lines 19-22; column 17, lines 10-52) (Fig. 5F), the data file comprising a point-of-failure indication within the failed routine (Fig. 5F: column 17, lines 17-21), parameters of the failure (column 17, lines 35-43), an identifier of the associated electronic page (columns 17-18: lines 62-l2) (Fig. 5F: "URL: www.rnercint.com"), and stores the data file in the data repository sending notification of the action to the developer (column 2, lines 39-40; column 6, lines 15-23).

It would have been obvious to one of ordinary skill in the art at the time of the invention to have stored the failed navigation script of DaCosta and for the proper notifications of the failed script to have included a point in process of the failure along with an identifier of the associated web page, because Weinberg teaches that by storing the failed navigation script, a developer can easily display the results of the navigation and quickly determine the location of the failure of the routine (column 3, lines 29-44). This would have made the re-teaching (i.e. correcting) of the navigation script easier for the developer (column 6, lines 9-13 & 35-41; column 18, lines 42-67).

Applicant's Response

Applicant herein amends claims 1 and 12 to include further limitations on the claimed functional logic blocks. Applicant reproduces claim 1 below as an aid in discussion:

1. (Currently amended) A software application executing on a server for enabling automated notification of applied structural changes to electronic information pages

hosted on a data packet network comprising:

a developer-interface module for enabling developers to build and modify network navigation and interaction templates using functional logic blocks, for automatically navigating to and interacting with interactive electronic information pages on the data packet network;

a navigation system-interface module for integrating the software application to a proxy-navigation system for periodic execution of the templates;

a change-notification module for indicating a point in process where a navigation and interaction routine has failed and for creating a data file containing parameters associated with the failed routine; and

a database interface module for interfacing the software application to a data repository for storing the data file;

wherein the functional logic blocks are modular parts of the navigation and interaction templates containing all of the possible navigation and interaction instructions required by the navigation system-interface module and the software application periodically submits test navigation and interaction routines to the navigation system for execution by virtue of the interface with the navigation system, and upon failure of a test routine, creates the data file, the data file comprising a point-of-failure indication within the failed routine identifying the functional logic block in the associated template, an identifier of the associated electronic information page subjected to the navigation routine, and stores the data file in the data repository sending notification of the action to the developer.

Claim1 now recites that the functional logic blocks are modular parts of the navigation and interaction templates containing all of the possible navigation and interaction instructions required by the navigation system-interface module and the software application periodically submits test navigation and interaction routines to the navigation system for execution by virtue of the interface with the navigation system, and upon failure of a test routine, creates the data file, the data file comprising a point-of-

failure indication within the failed routine by at least identifying the functional logic block in the associated template.

In applicant's invention the modular concept used in constructing navigation orders enables site-logic portions as well as login portions and other functional portions of the order to be interchanged in an efficient manner. For example, if a navigation order enables navigation to and auto-login with a particular website, the same order may be used for all individuals. Only the auto-login block of the order, which is different for each individual, would need to be interchanged. The inventor provides through software 243 that existing navigation orders may have obsolete site-logic blocks replaced with updated site-logic blocks in an automated fashion. More about this capability is described below. (page 61, lines 8-17)

Applicant points out that DaCosta teaches recording user interactions such as mouse clicks and hyperlink selection in order to play back for automatic electronic data collection. Specifically, DaCosta teaches:

"The navigation API 10 includes a recording module 12 and playback module 14. For example, if a web site requires a user to enter a login name and password to reach an orientation page and then asks for a set of preferences to go to specific web pages or other web- accessible documents of interest, it is an object of the present invention to enable a client application to record this path once, then play it back many times including the dialog interaction with the server." (col. 5, lines 43-51)

"Additionally, as shown in FIG. 1, an extraction API 20 enables an application to robustly define data segments in a web page or other web-accessible document. Similar to the case of the navigation module 10, there are recording and playback modules 22 and 24 included in the extraction API 20. However, instead of recording playing across web pages and web space, the extraction module 20 records and plays across elements within a single page." (col. 5, lines 55-63)

As clearly seen from the above portions, DaCosta records and plays back user

interactions in order to navigate to Web sites to collect user specified data. Applicant argues that Weinberg also provides a facility for recording user interactions.

In applicant's invention, modular functional logic blocks are used to build navigation templates, in this manner when navigation fails only replacement or repair of a logic block occurs, sometimes automatically, rather than the requirement of re-recording user interactions as in DaCosta and Weinberg.

Applicant believes claims 1 and 12, as amended, are easily patentable over the art of DaCosta and Weinberg. Dependent claims 2-11 and 13-17 are patentable on their own merits, or at least as depended from a patentable claim.

Examiner's Rejection

-In regard to independent claim 18, DaCosta teaches a method for receiving automated notification of random structural changes applied to electronic information pages hosted on a network comprising:

-establishing notification of a failed navigation and interaction routine executed for the purpose of navigating to and interacting with an electronic information page (column 6, lines 9-13 & 35-41; column 18, lines 34-67: "email or pager notification"). -creating an instance of the failed routine associated with the cause of failure (column 18, lines 43-67: "it is known the script has failed. . . and proper notifications sent to individuals or entities responsible for the operation of the failing script by email.. . for example"; column 19, lines 1-15);

-accessing the notification of the of the failed routine for review purposes (column 6, lines 9-13 & 35-41; column 18, lines 34-67: i.e. developer accesses failed script for reteaching purposes);

-being able to navigate to the electronic information page identified in the recorded instance (column 6, lines 9-13 & 35-41; column 18, lines 34-67: i.e. developer accesses failed script for re-teaching purposes);

-accessing source information associated with the electronic information page identified in the recorded instance (i.e. re-teaching a new navigation and extraction script

by accessing the source information).

-creating new logic according to the source information and according to information contained in the recorded instance (column 6, lines 9-13 & 35-41; column 18, lines 34-67);

installing the new logic into existing navigation templates that depend on the updated information for successful function (column 6, lines 9-13 & 35-41; column 18, lines 34-67; column 19, lines 1-15).

DaCosta does not specifically teach wherein the instance of the failed navigation routine was stored for future review including parameters associated with the failed routine. Weinberg teaches storing the data file (column 2, lines 39-40; column 6, lines 19-22), wherein the application periodically submits test navigation and interaction routines (column 6, lines 19-22), and upon failure of the routine, creates a data file (column 2, lines 39-40; column 3, lines 29-43; column 6, lines 19-22; column 17, lines 10-52) (Fig. 5F), the data file comprising a point-of-failure indication within the failed routine (Fig. SF: column 17, lines 17-21), parameters of the failure (column 17, lines 35-43), an identifier of the associated electronic page (columns 17-18: lines 62-12) (Fig. 5F: "URL: www.mercint.com"), and stores the data file in the data repository sending notification of the action to the developer (column 2, lines 39-40; column 6, lines 15-23).

It would have been obvious to one of ordinary skill in the art at the time of the invention to have stored the failed navigation script of DaCosta and for the proper notifications of the failed script to have included a point in process of the failure along with an identifier of the associated web page, because Weinberg teaches that by storing the failed navigation script, a developer can easily display the results of the navigation and quickly determine the location of the failure of the routine (column 3, lines 29-44). This would have made the re-teaching (i.e. correcting) of the navigation script easier for the developer (column 6, lines 9-13 & 35-41; column 18, lines 42-67).

Applicant's Response

Claim 18 is also amended as in claims 1 and 12 as follows:

18. (Currently amended) A method for receiving notification of random structural changes applied to electronic information pages accessed by a proxy network navigation and interaction system and effecting updates to navigation templates based on the change information comprising steps of:

(a) establishing notification of a failed navigation and interaction routine executed for the purpose of navigating to and interacting with an electronic information page on a data-packet-network;

(b) recording an instance of the failed routine including parameters associated with the cause of failure identification of at least one a plurality of modular logic blocks used to build the navigation templates;

(c) accessing the recorded instance of the failed routine for review purposes;

(d) navigating to the electronic information page identified in the recorded instance on the data-packet-network;

(e) accessing source information associated with electronic information page identified in the recorded instance;

(f) creating Withdrawn creating a new modular logic block according to the source information and according to information contained in the recorded instance; and

(g) installing the Withdrawn newly created modular logic block into existing navigation templates that depend on the updated information for successful function.

Applicant argues that method claim 18, as amended, is also patentable over the art of DaCosta and Weinberg as argued above on behalf of claims 1 and 12. Dependent claims 19-28 are patentable on their own merits, or at least as depended from a patentable claim.

Summary

Applicant claims an ability to build and modify network navigation and interaction templates (navigation orders) using functional logic blocks, meaning software logic blocks including function, not recorded portions of navigation as in DaCosta and

Weinberg. The art of DaCosta and Weinberg fails to teach creating navigation scripts using logic blocks, as claimed. In applicant's invention a site logic portion of a navigation script may contain more than one identifiable interaction task. Therefore, site-logic blocks, which are modular parts of whole navigation orders contain all of the possible interaction instructions available at the associated site. In a given navigation order, the associated site-logic blocks are activated to enable only the specified interactions described in the request portion of the order specified by the requesting user. The modular concept used in constructing navigation orders enables site-logic portions as well as login portions and other functional portions of the order to be interchanged in an efficient manner. Applicant argues, as evidenced above, DaCosta fails to teach using functional logic blocks, as claimed.

Based on the above claim amendments and arguments, applicant believes the claims are patentable over the art of DaCosta and Weinberg.

If there are any time extensions needed beyond any extension specifically requested with this amendment, such extension of time is hereby requested. If there are any fees due beyond any fees paid with this amendment, authorization is given to deduct such fees from deposit account 50-0534.

Respectfully Submitted,
Tim Armandpour et al.

By */Donald R. Boys/*
Donald R. Boys
Reg. No. 35,074

Central Coast Patent Agency, Inc.
3 Hangar Way, Suite D
Watsonville, CA 95076
831-768-1755